Help Center

Binary patterns in NAND flash memory

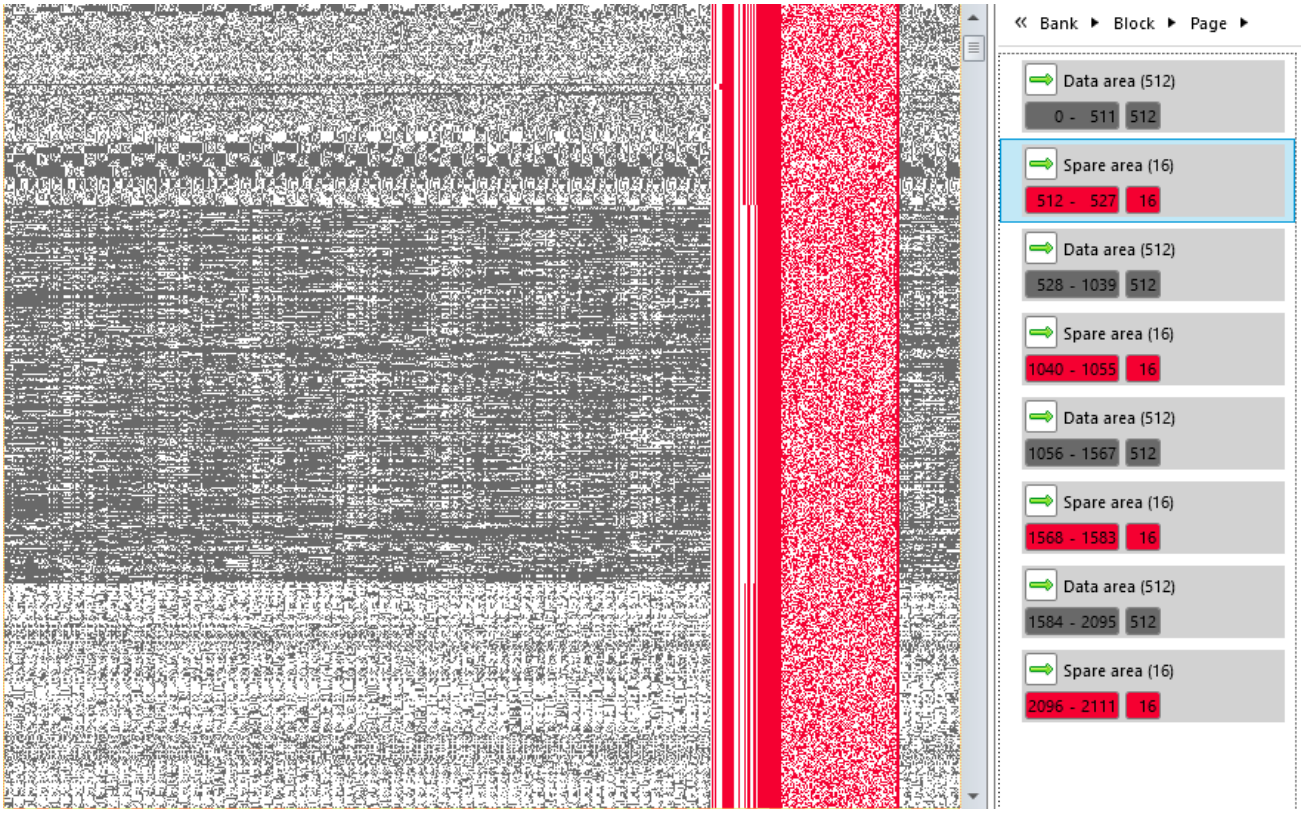
Analysis and recognition of binary patterns in NAND flash memory is the key step in chip-off data recovery and digital forensic analysis of broken flash devices. This analysis is carried out in the Bitmap mode since the classic HEX view does not allow to trace binary patterns. To show patterns in Bitmap mode properly, the correct page size should be set.

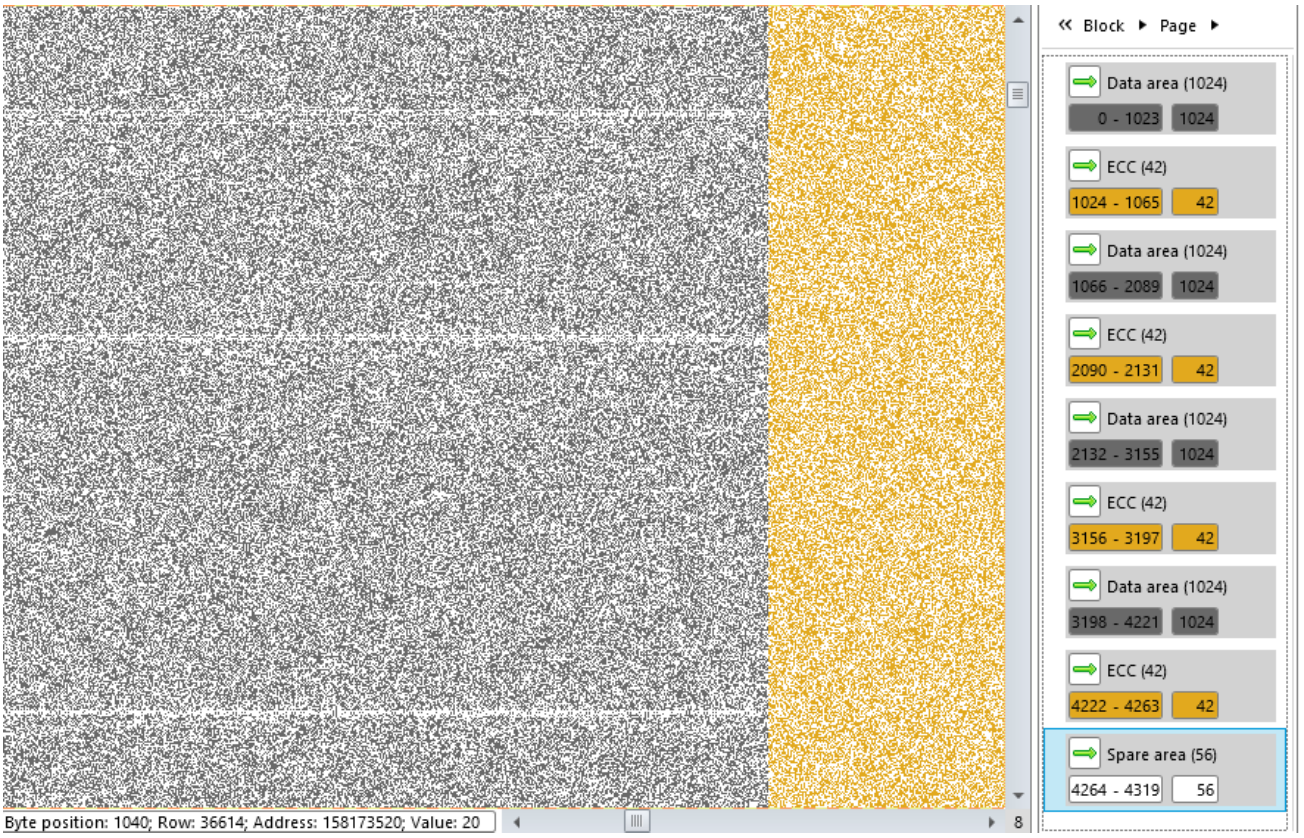
The Physical image of NAND contains the following recognizable patterns:

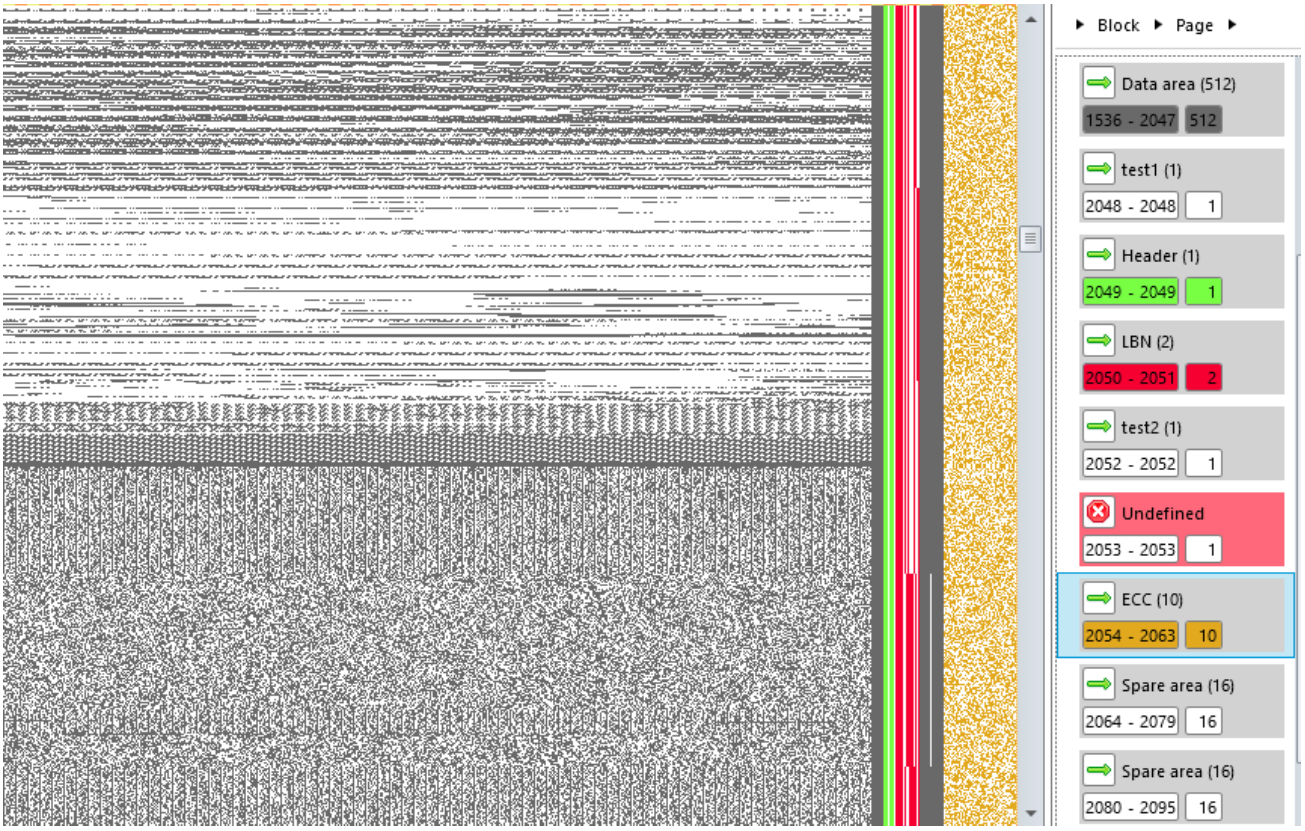
- Data area patterns
- Scrambled (XORed) data area patterns
- Spare area patterns
- ECC patterns
- Bit error patterns
- Bad Columns patterns
- Other patterns

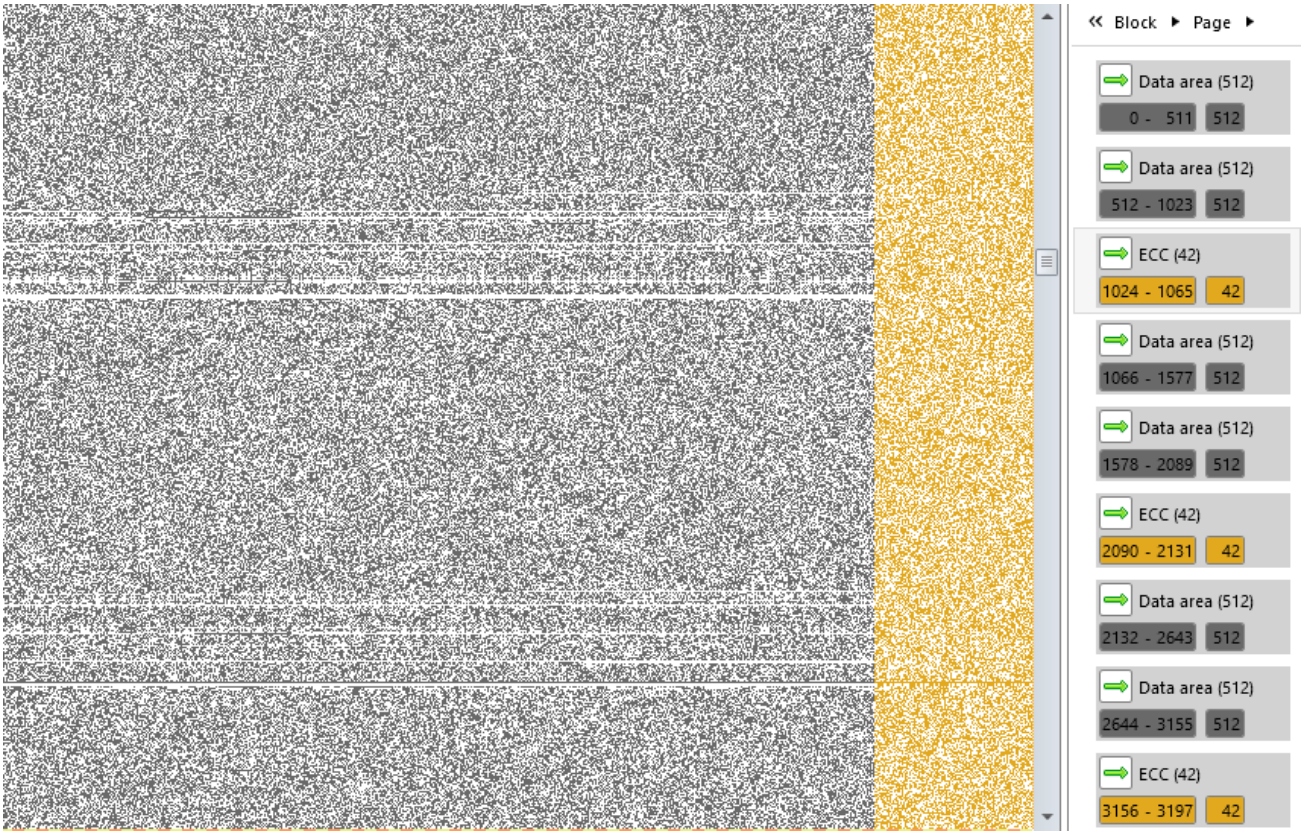
Data area

The Data has a quiet high density (entropy) and has horizontal patterns that look like stripes, cubes, etc. The pictures below represent typical patterns of sectors with data (Data area) within pages. The Data area is highlighted in grey.


Bank
Block
Page
Data area (512)
0 - 511 512
Spare area (16)
512 - 527 16
Data area (512)
528 - 1039 512
Spare area (16)
1040 - 1055 16
Data area (512)
1056 - 1567 512
Spare area (16)
1568 - 1583 16
Data area (512)
1584 - 2095 512
Spare area (16)
2096 - 2111 16


Block
Page
Data area (1024)
0 - 1023 1024
ECC (42)
1024 - 1065 42
Data area (1024)
1066 - 2089 1024
ECC (42)
2090 - 2131 42
Data area (1024)
2132 - 3155 1024
ECC (42)
3156 - 3197 42
Data area (1024)
3198 - 4221 1024
ECC (42)
4222 - 4263 42
Spare area (56)
4264 - 4319 56
Byte position: 1040; Row: 36614; Address: 158173520; Value: 20
8



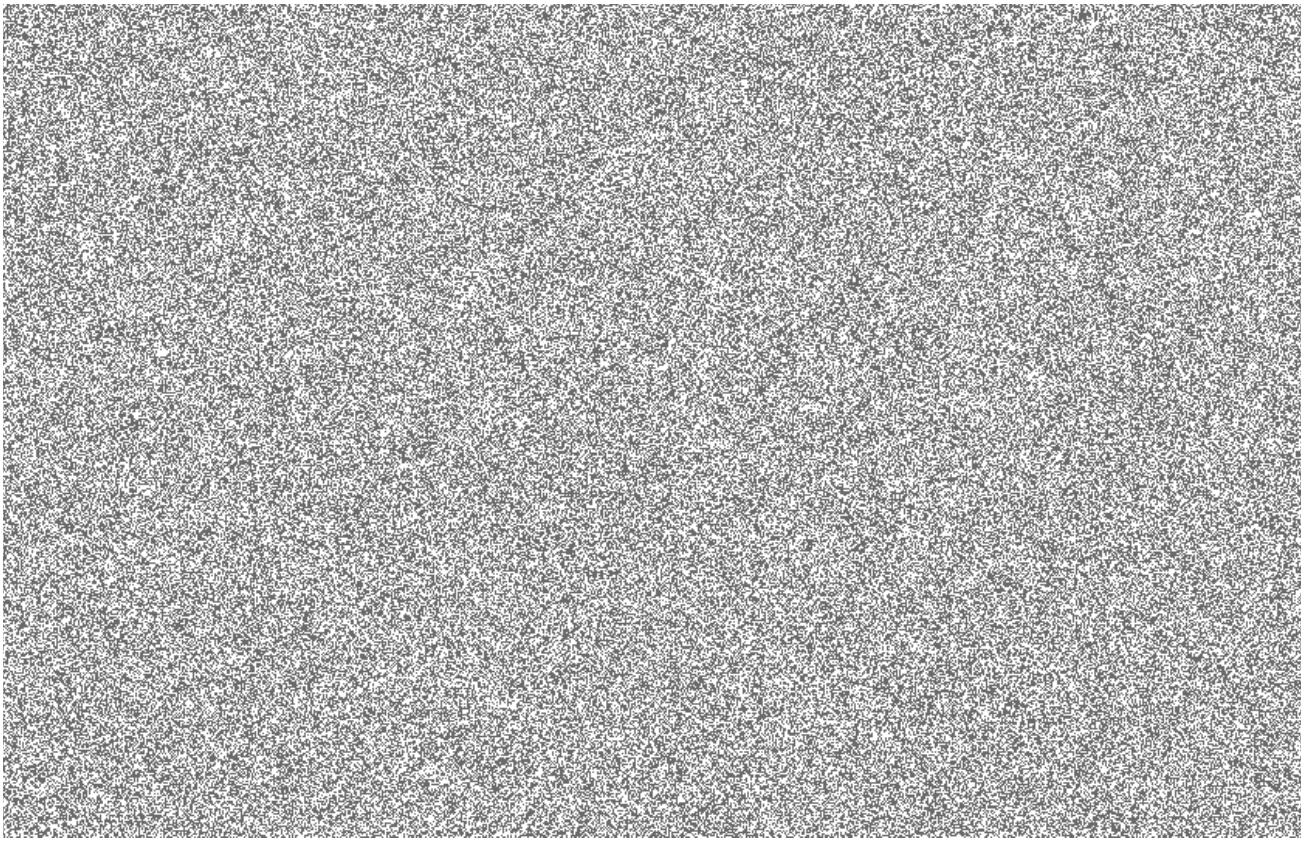


The determination of horizontal boundaries of data patterns helps to determine the structure of the page.

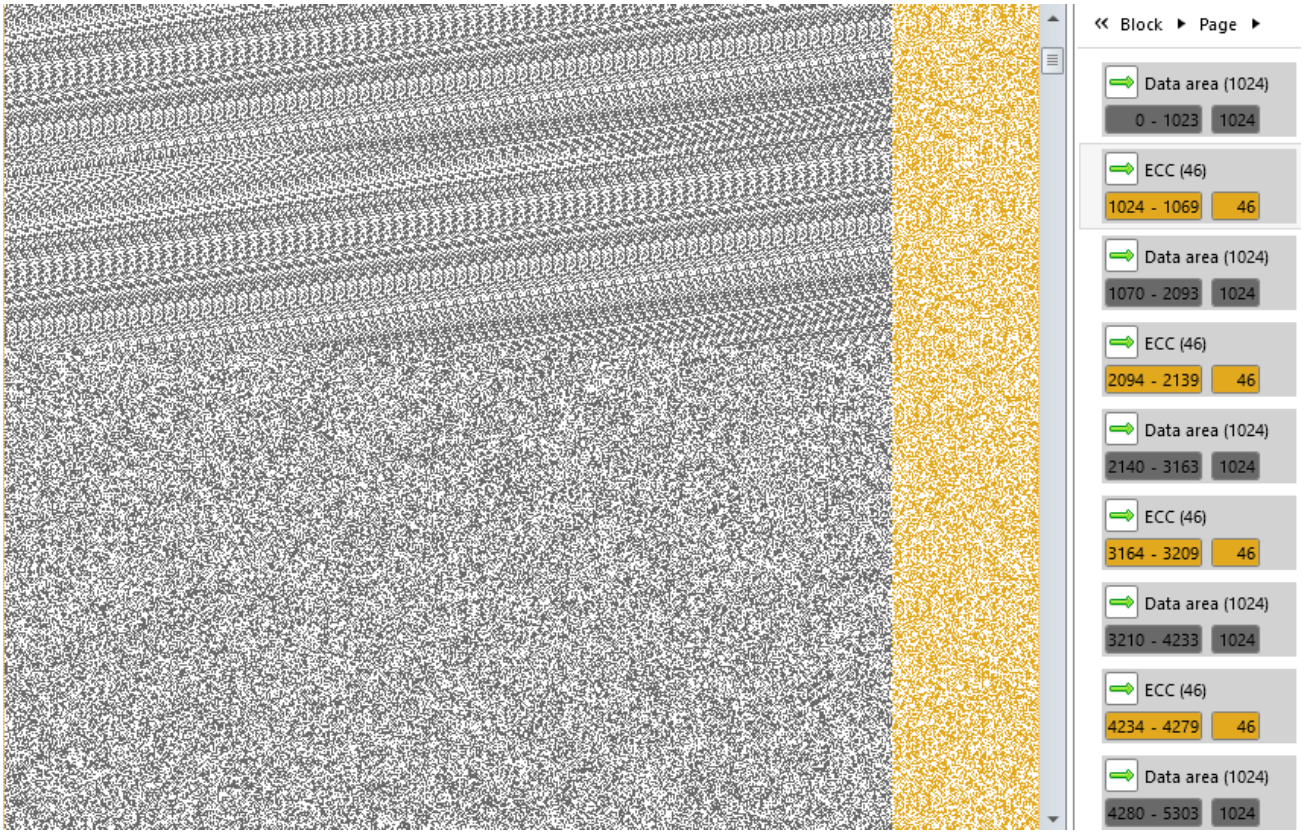
Scrambled (XORed) data

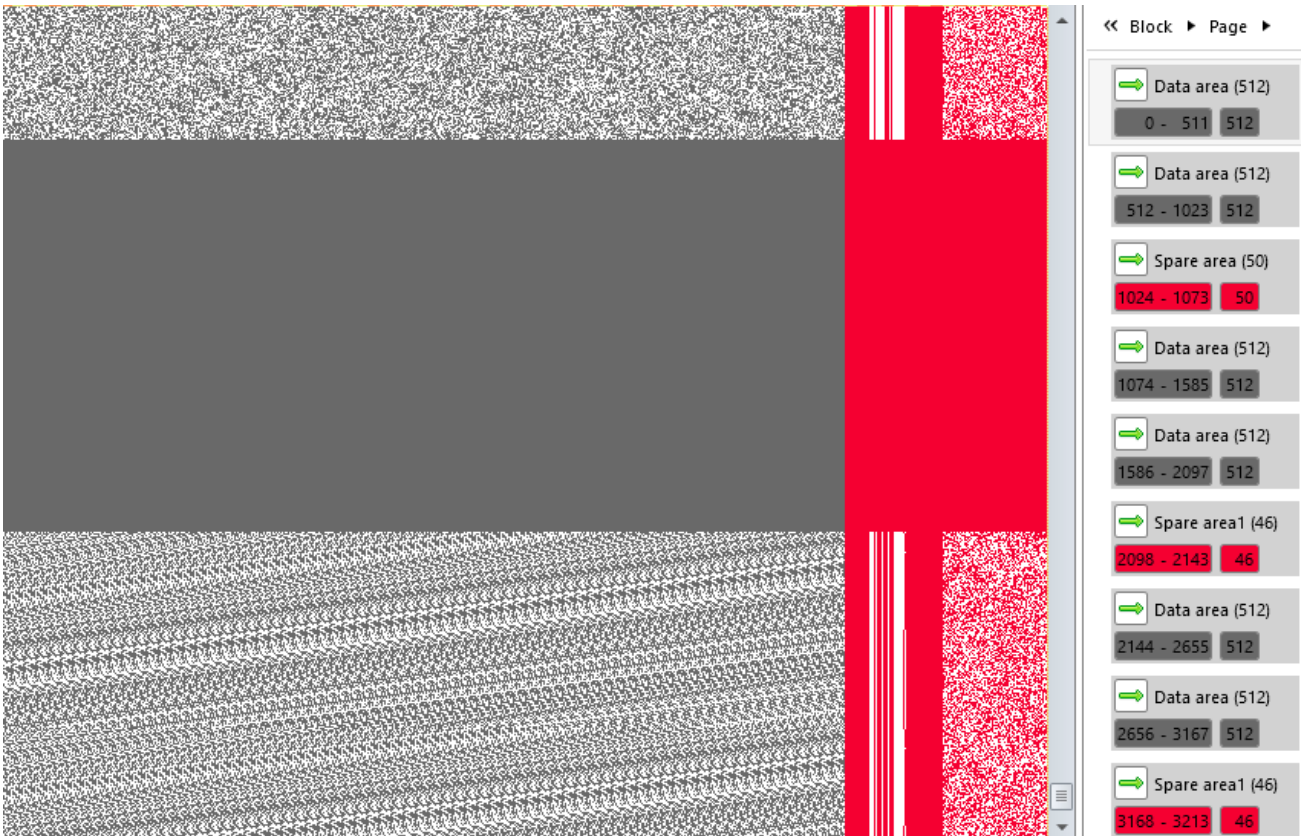
Scrambled (XORed) data has no typical patterns of data. It looks like a

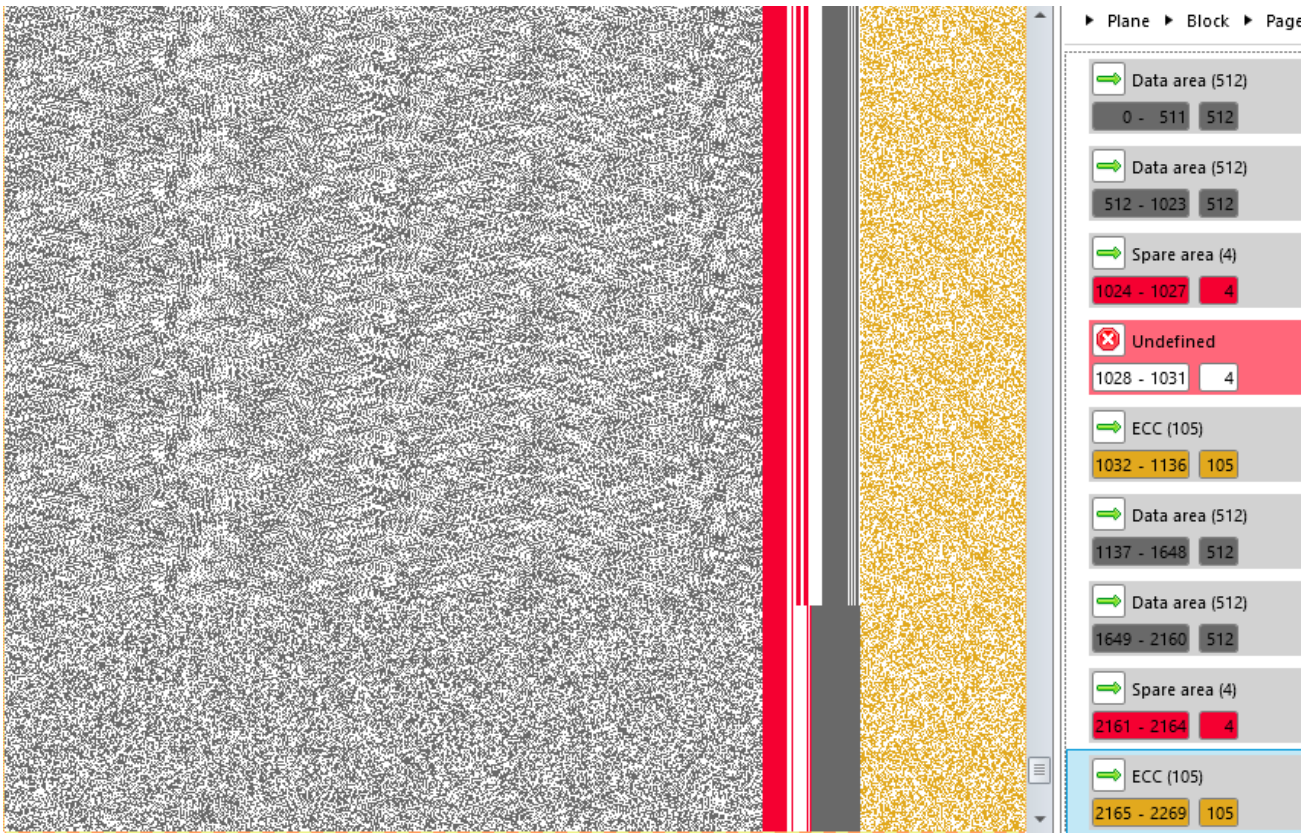
noise with very high density (entropy of white noise).

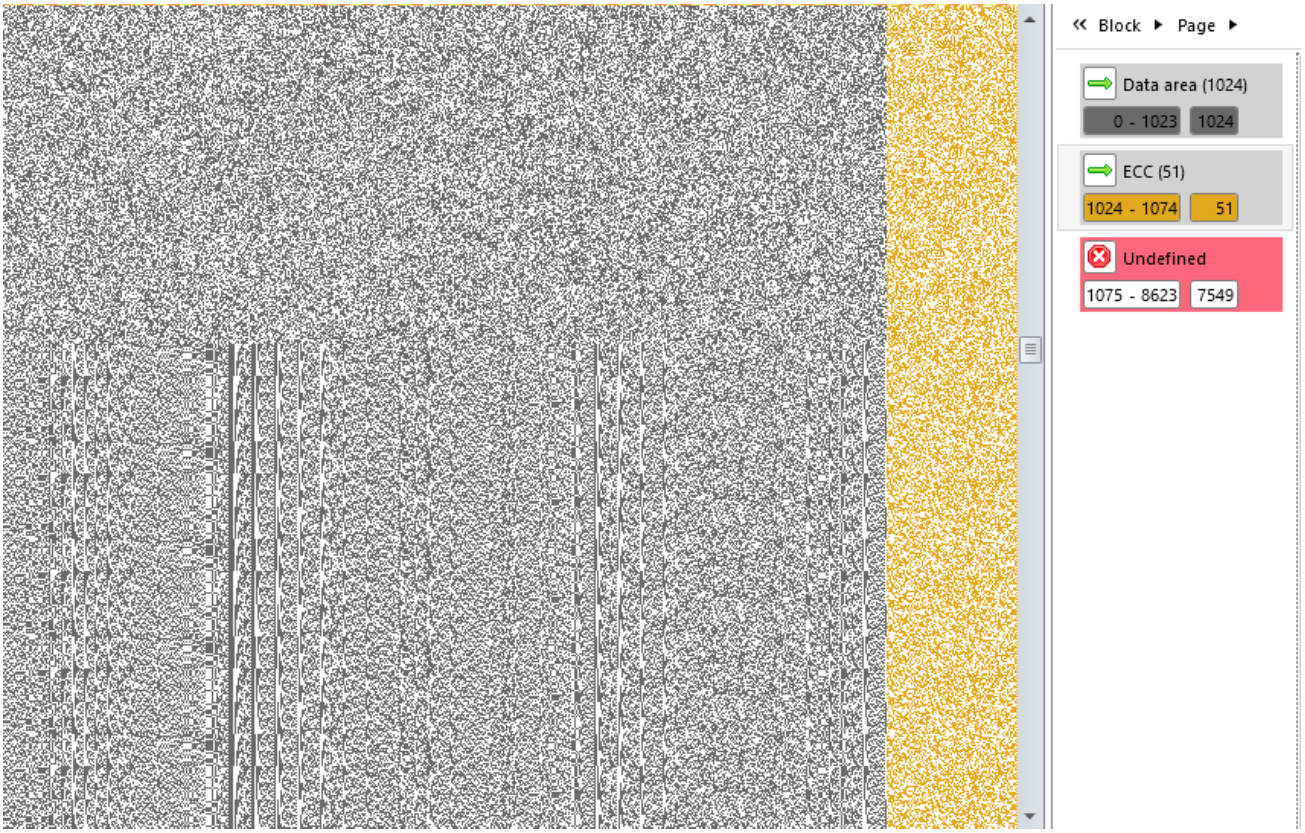


However, when scrambling performed on a data area that contains zeros, there are very specific patterns appear in the noise, which belong to fragments of the XOR key. So, any pattern in the scrambled data area that doesn’t look like a noise is xor key fragments.




Block
Page
Data area (512)
0 - 511 512
Data area (512)
512 - 1023 512
Spare area (50)
1024 - 1073 50
Data area (512)
1074 - 1585 512
Data area (512)
1586 - 2097 512
Spare area1 (46)
2098 - 2143 46
Data area (512)
2144 - 2655 512
Data area (512)
2656 - 3167 512
Spare area1 (46)
3168 - 3213 46


Plane
Block
Page
Data area (512)
0 - 511 512
Data area (512)
512 - 1023 512
Spare area (4)
1024 - 1027 4
Undefined
1028 - 1031 4
ECC (105)
1032 - 1136 105
Data area (512)
1137 - 1648 512
Data area (512)
1649 - 2160 512
Spare area (4)
2161 - 2164 4
ECC (105)
2165 - 2269 105



Search of blocks with a large number of scrambled zeros (fragments of XOR key) is needed to determine the page structure and also to find a proper XOR key to decrypt data. The library of XOR patterns of different controllers is here.

Spare area

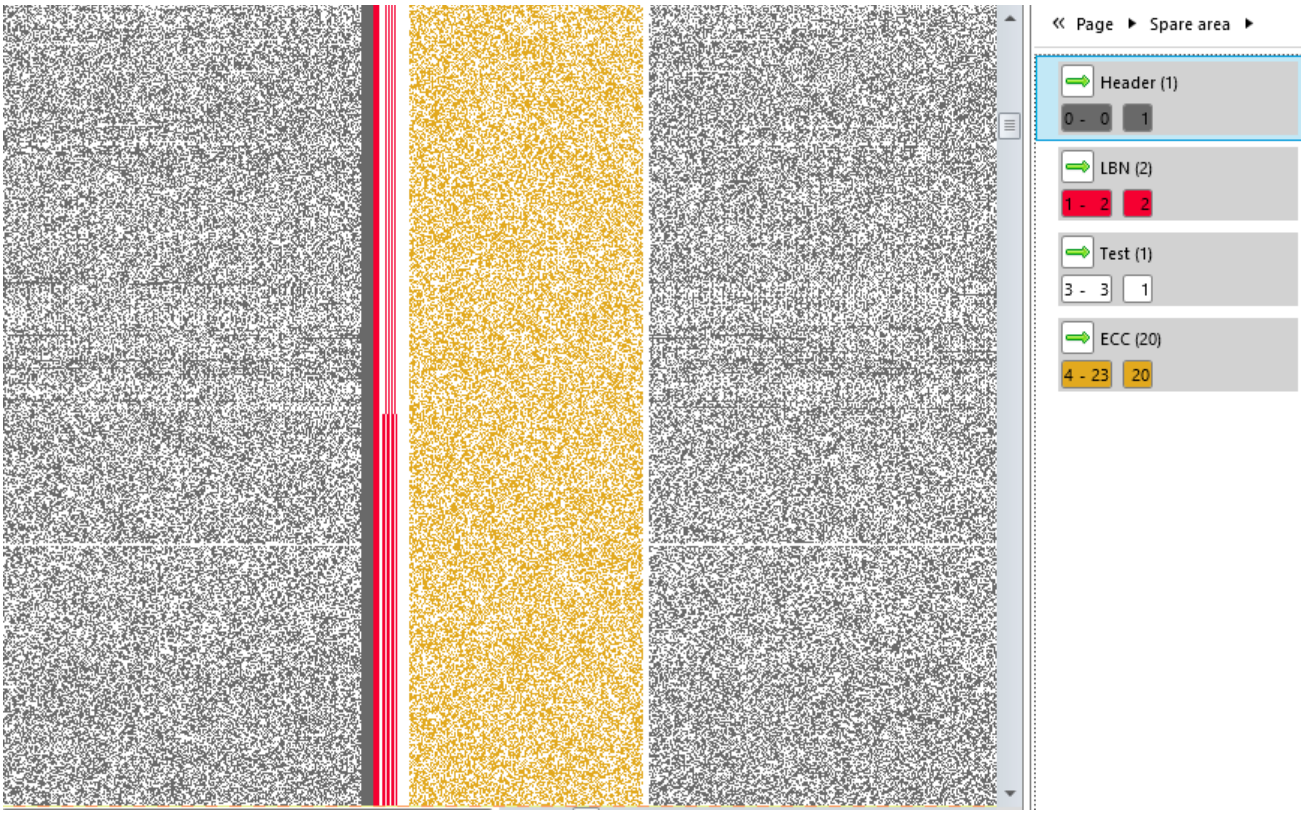
The spare area consists of bytes that define a position, type and other parameters of physical blocks, ECC, and others. In general, Spare area consists of the following patterns:

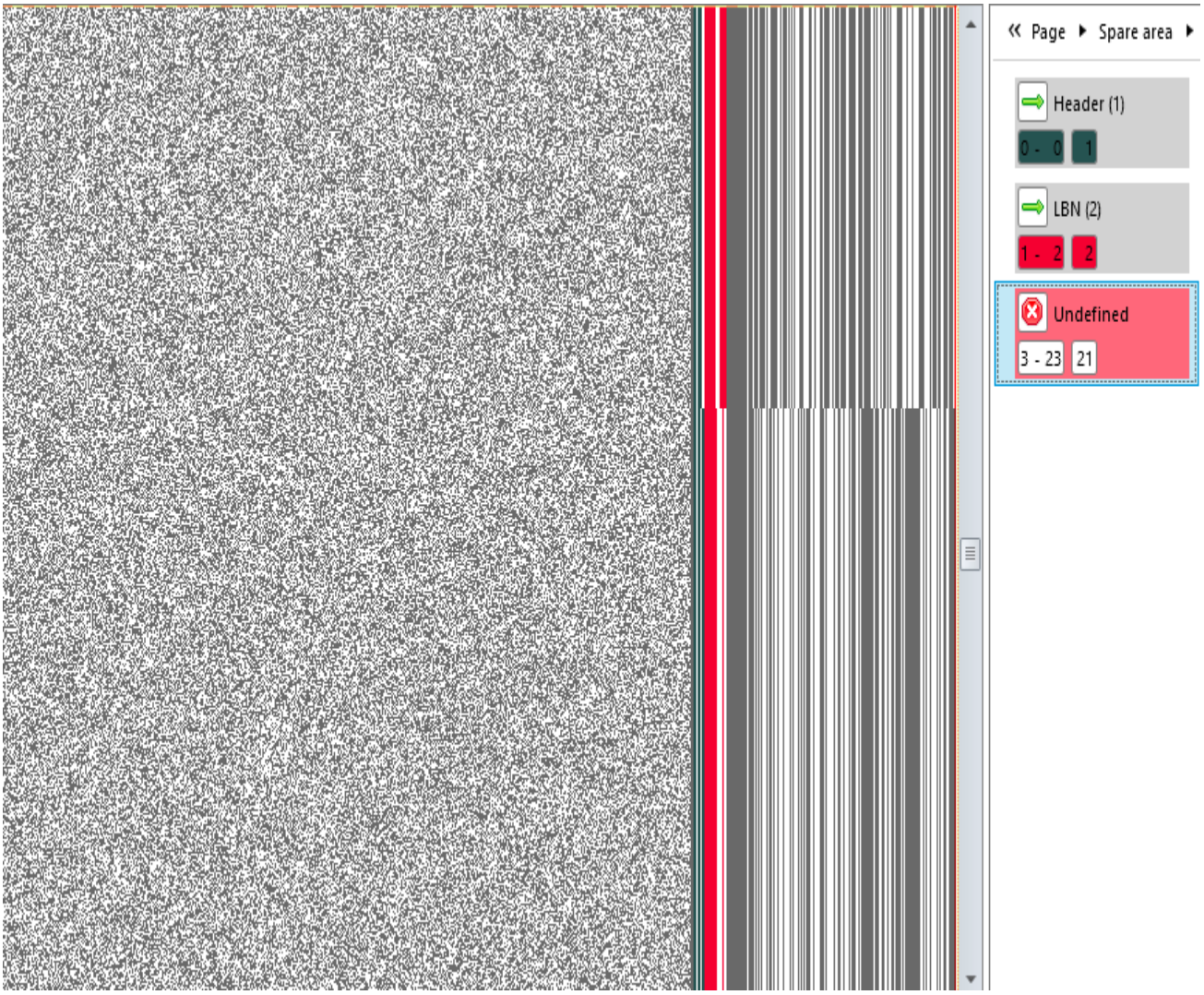
- LBN
- LPN
- Header
- ECC
- Bank Number
- Write counter

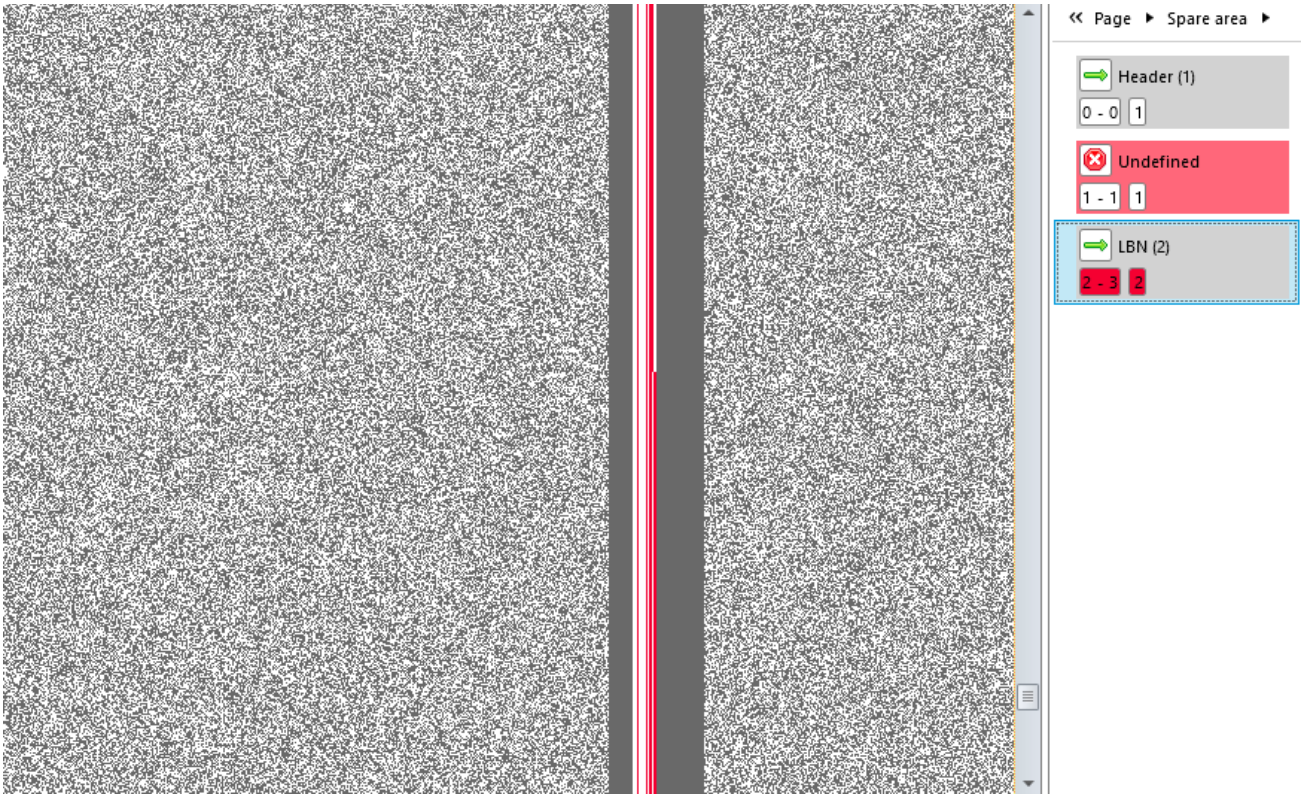
Spare area patterns represent vertical stripes, periodically changed from block to block.

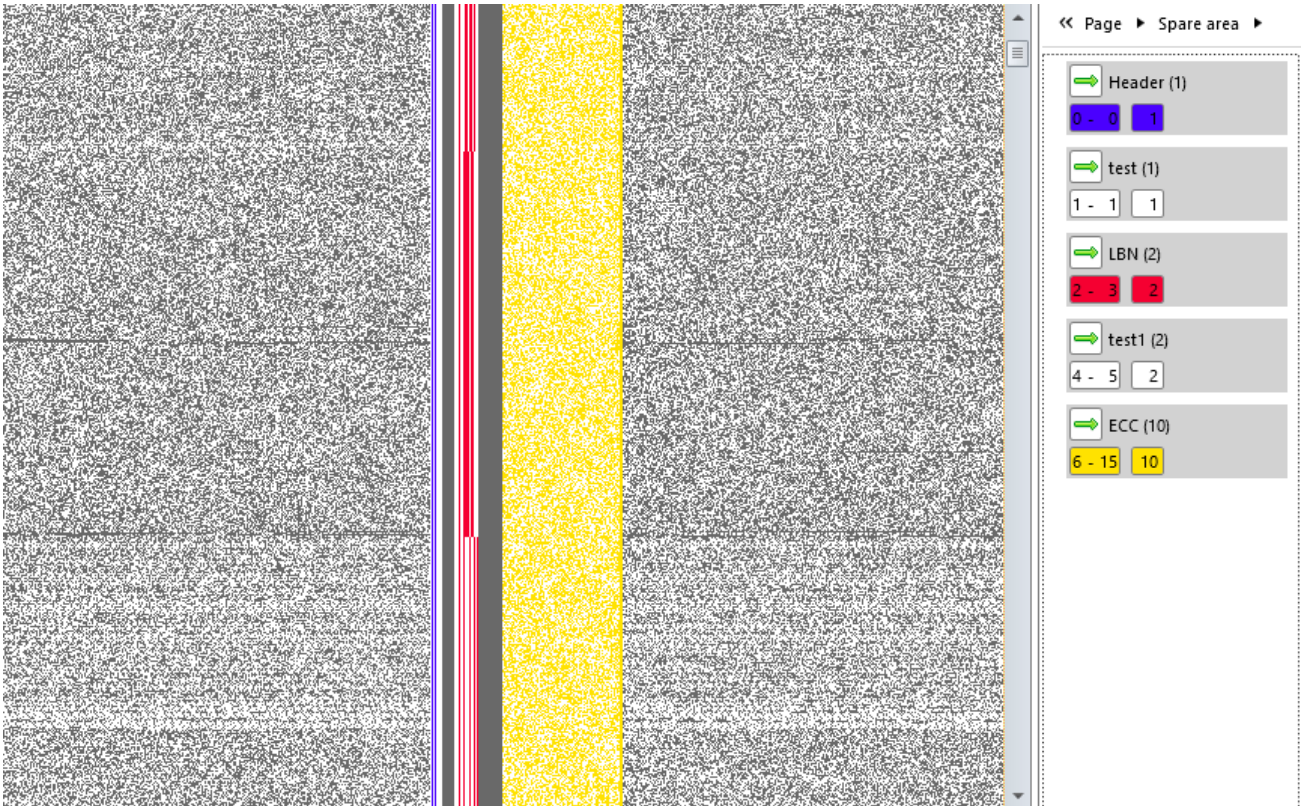
LBN

The logical block number pattern has 2 bytes (16 bits) size and being changed from block to block. It’s represented on the following pictures.




Page
Spare area
Header (1)
0 - 0
1
LBN (2)
1 - 2
2
Undefined
3 - 23
21

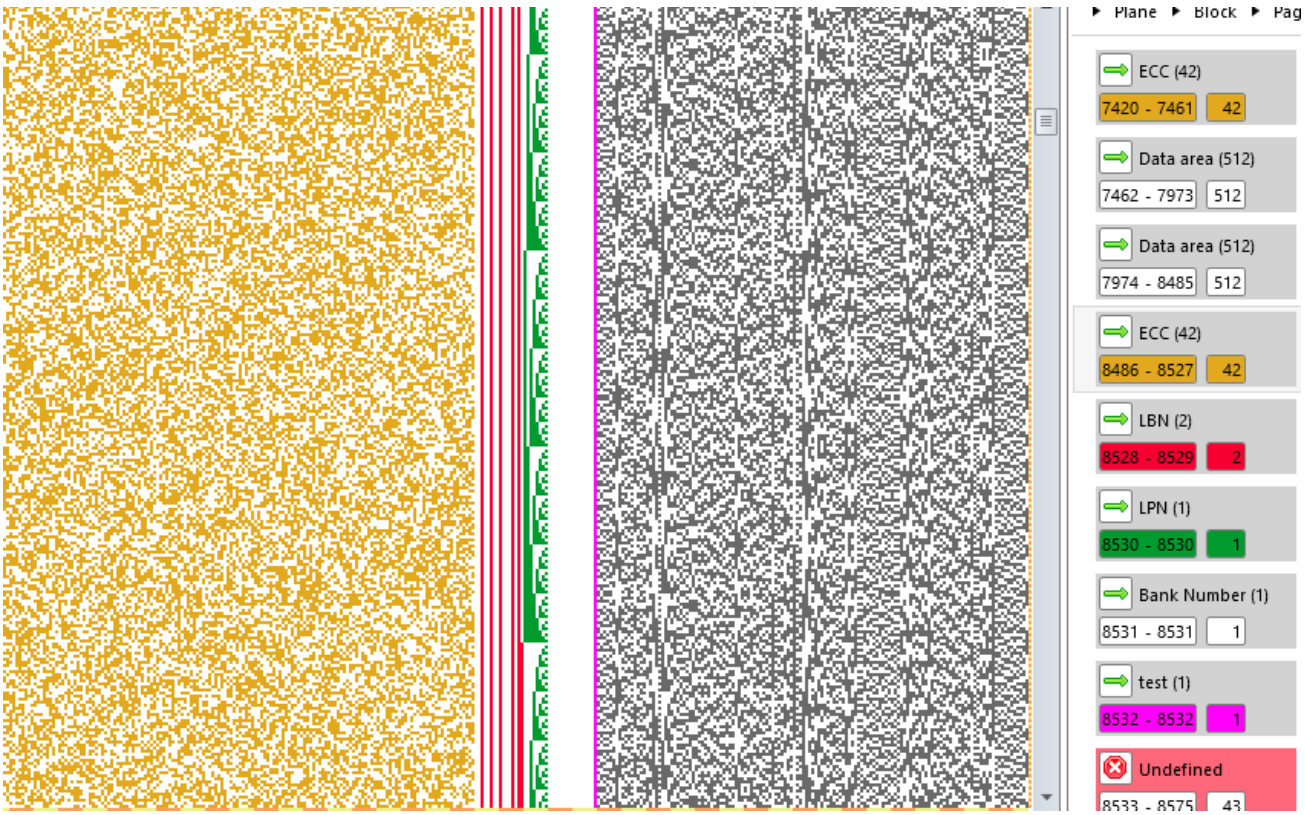

Page
Spare area
Header (1)
0 - 0
1
Undefined
1 - 1
1
LBN (2)
2 - 3
2

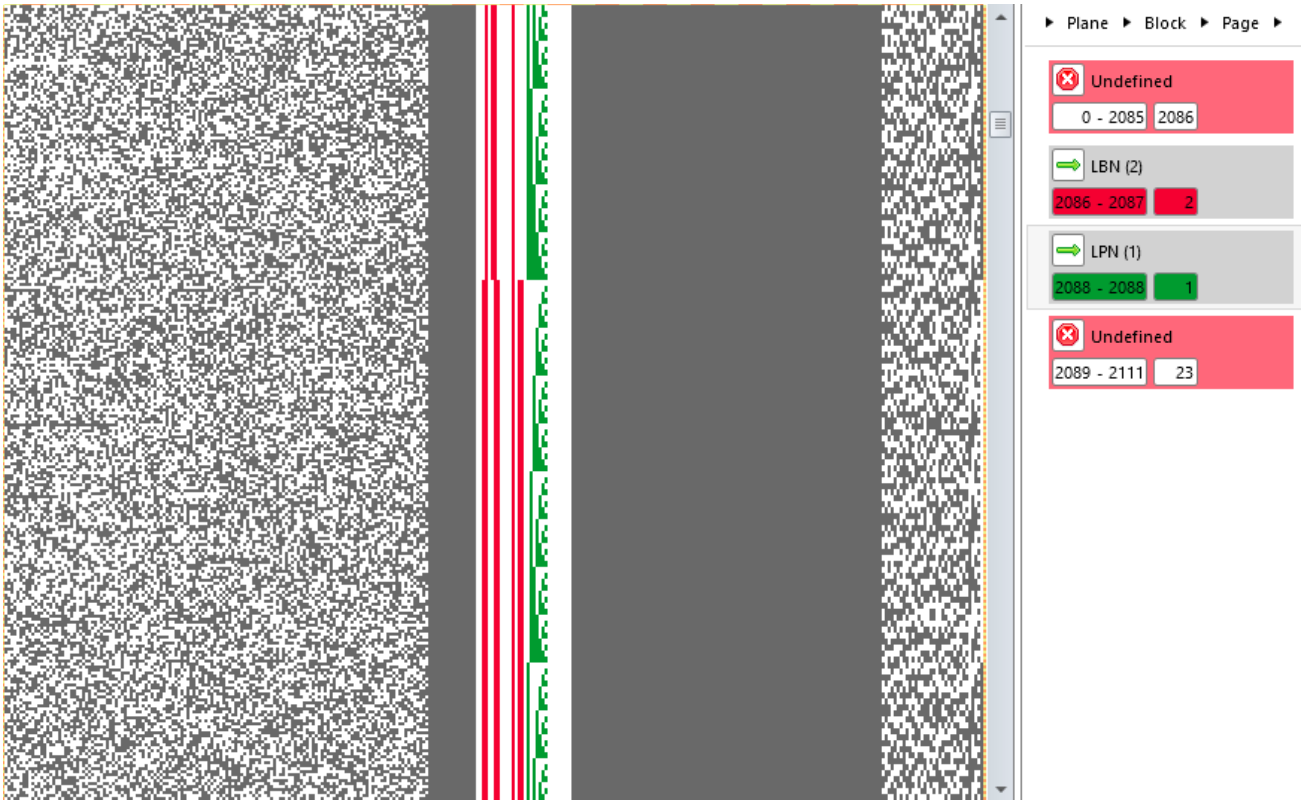


The places where LBN pattern changed are the boundaries of the virtual block.
The LBN pattern and it’s offset identification is essential for block arrangement into logical image (Markers table element settings).

LPN

Logical page number pattern has 1 or 2 bytes size and being changed from page to page within virtual block. It’s represented on following pictures.

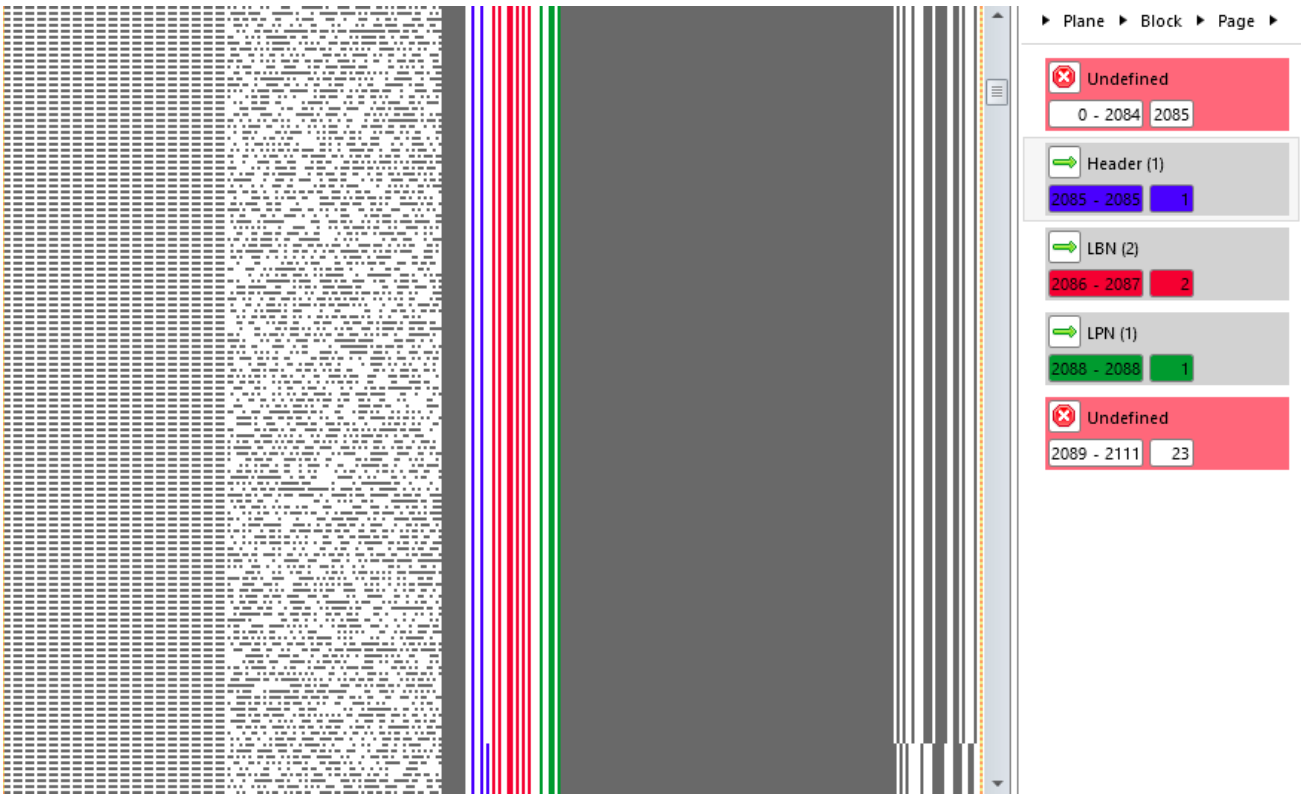


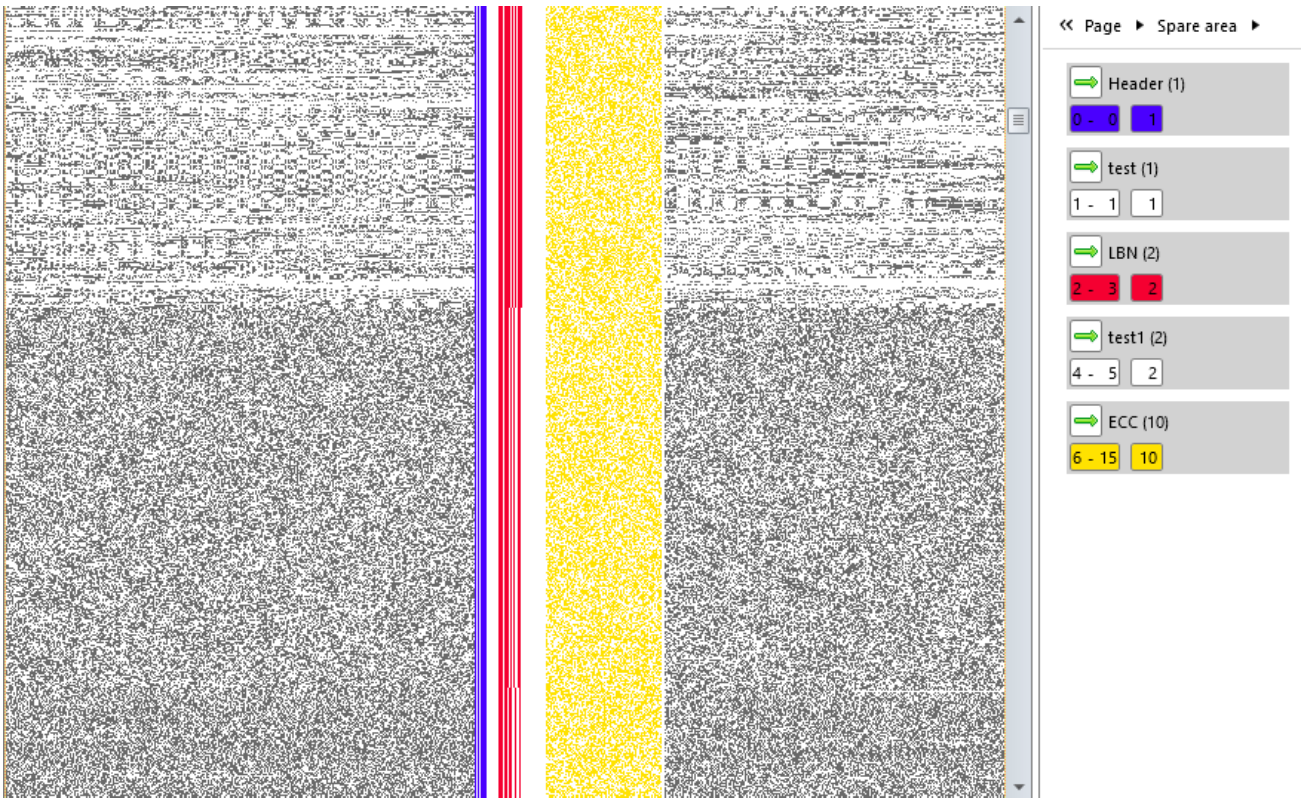


The LPN pattern and it’s offset identification is essential for block arrangement into logical image (Markers table element settings). More than 90% of controllers do not use LPN due to sequential page allocation within block.

Header

The block header pattern has 1-byte size and being changed very seldom, because 95-99% of all blocks in NAND are used to store data (main blocks) and have same header. It’s represented on the following pictures.

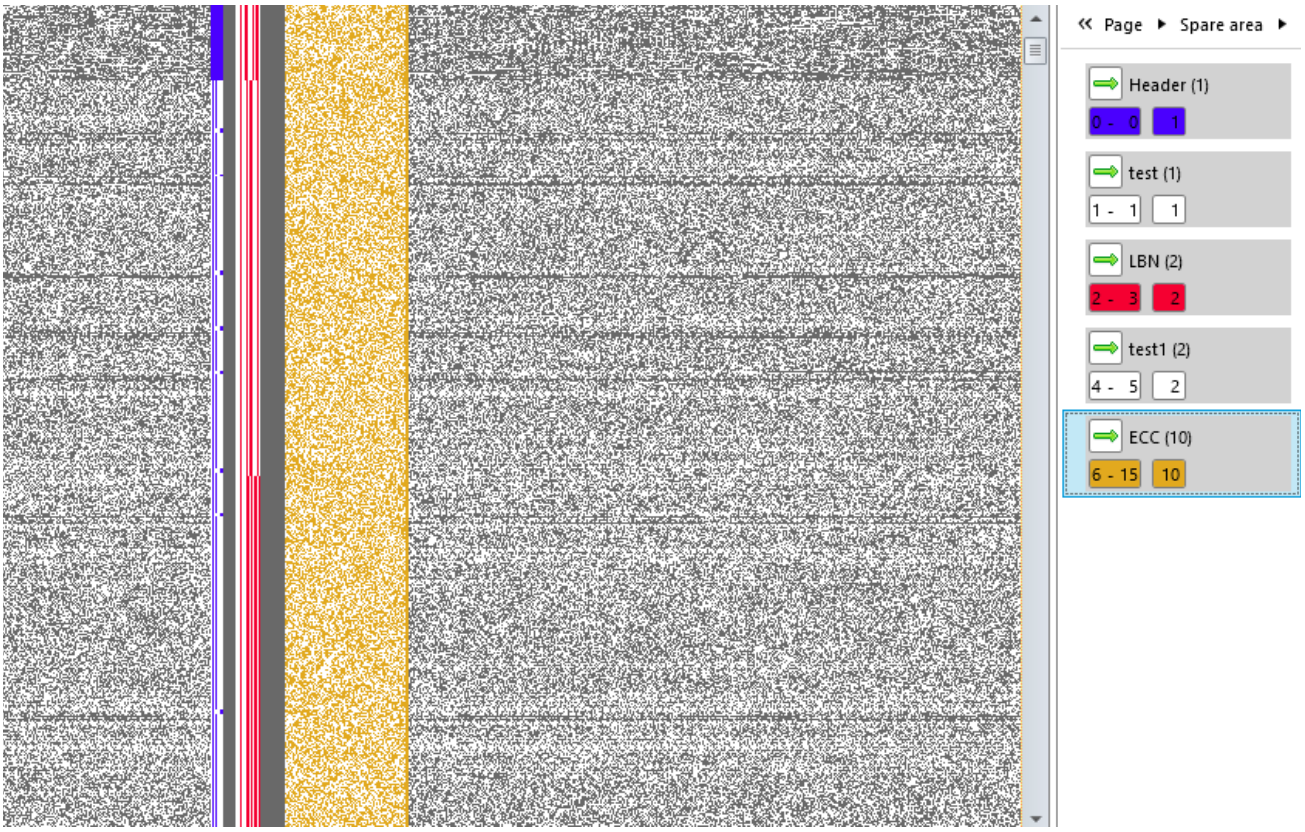


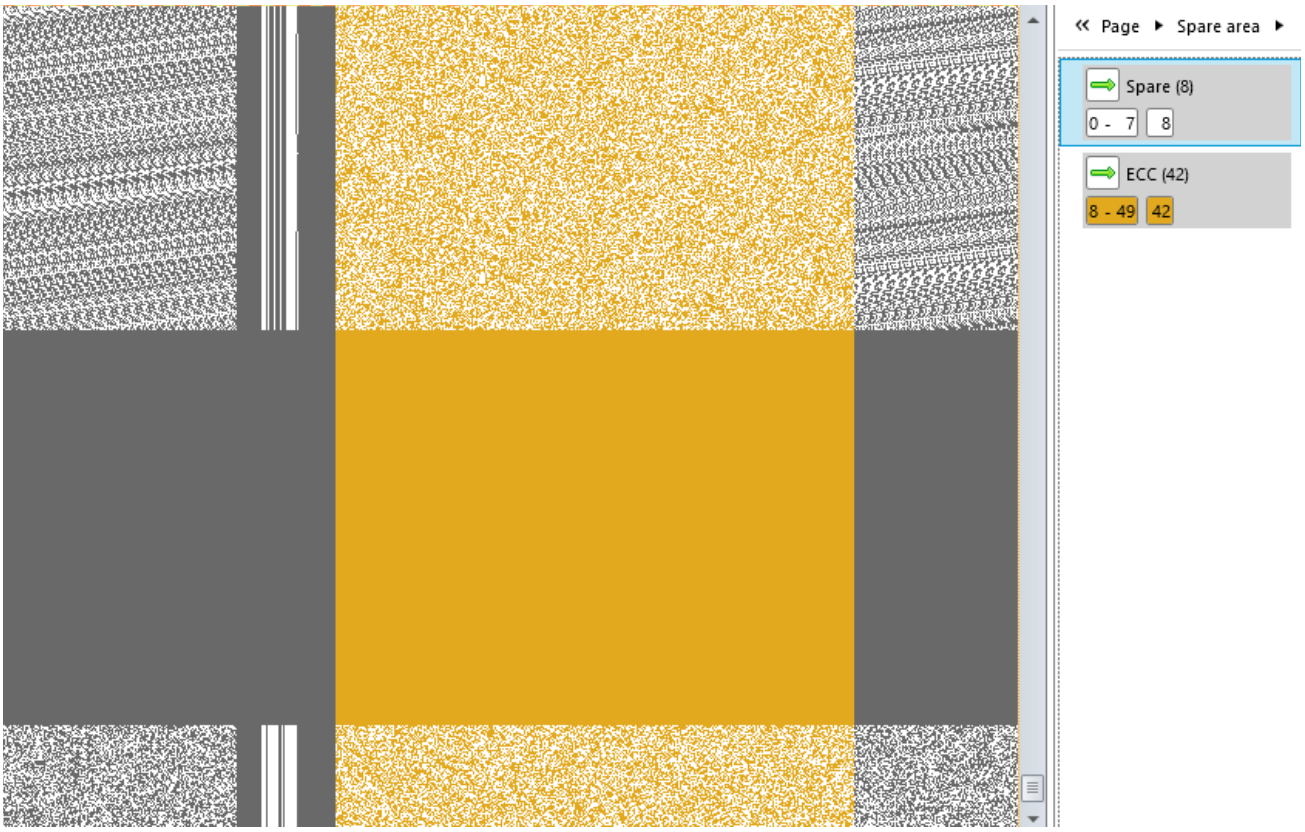


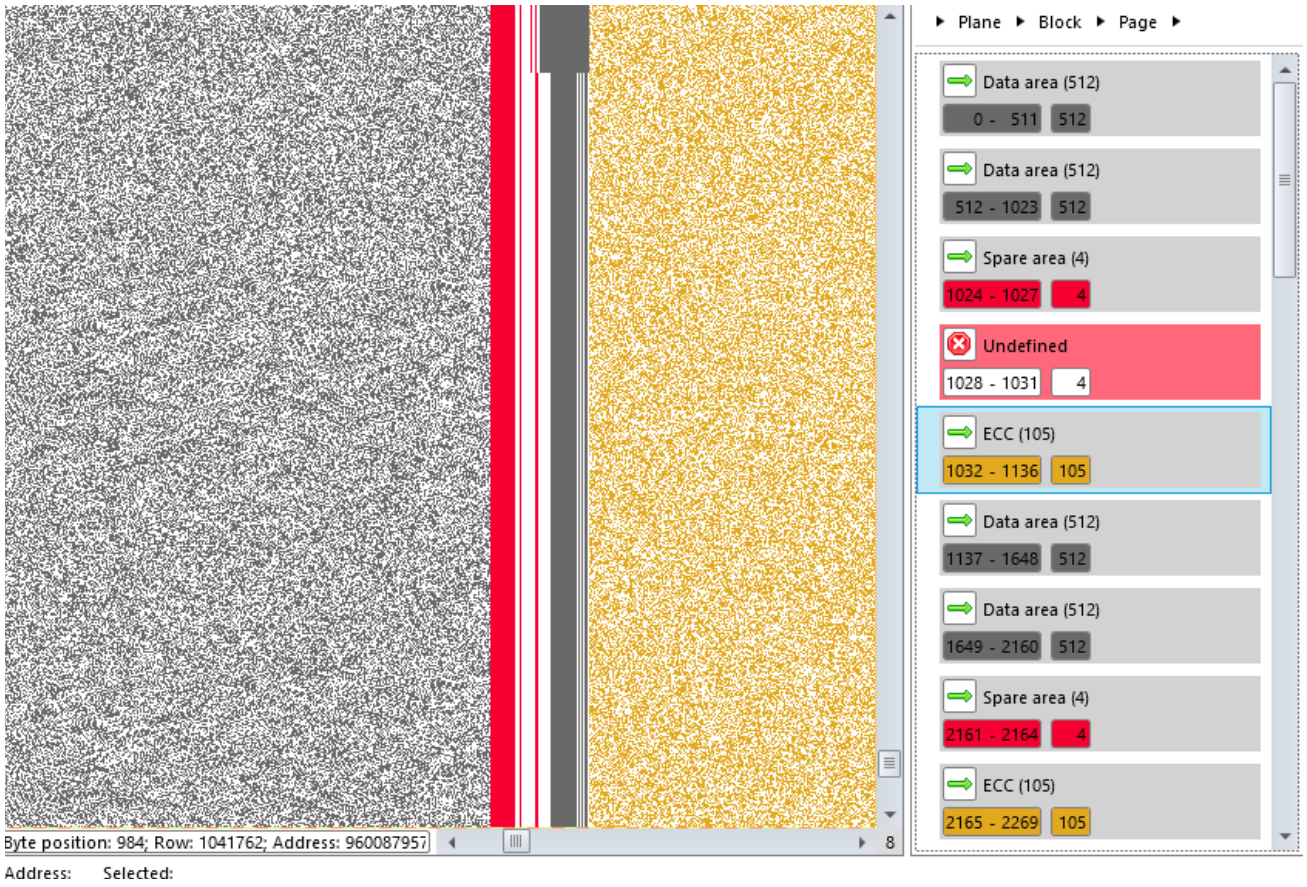
The Header pattern and it’s offset identification is essential for block filtering and sorting in the translation table when building logical image (Markers table element settings).

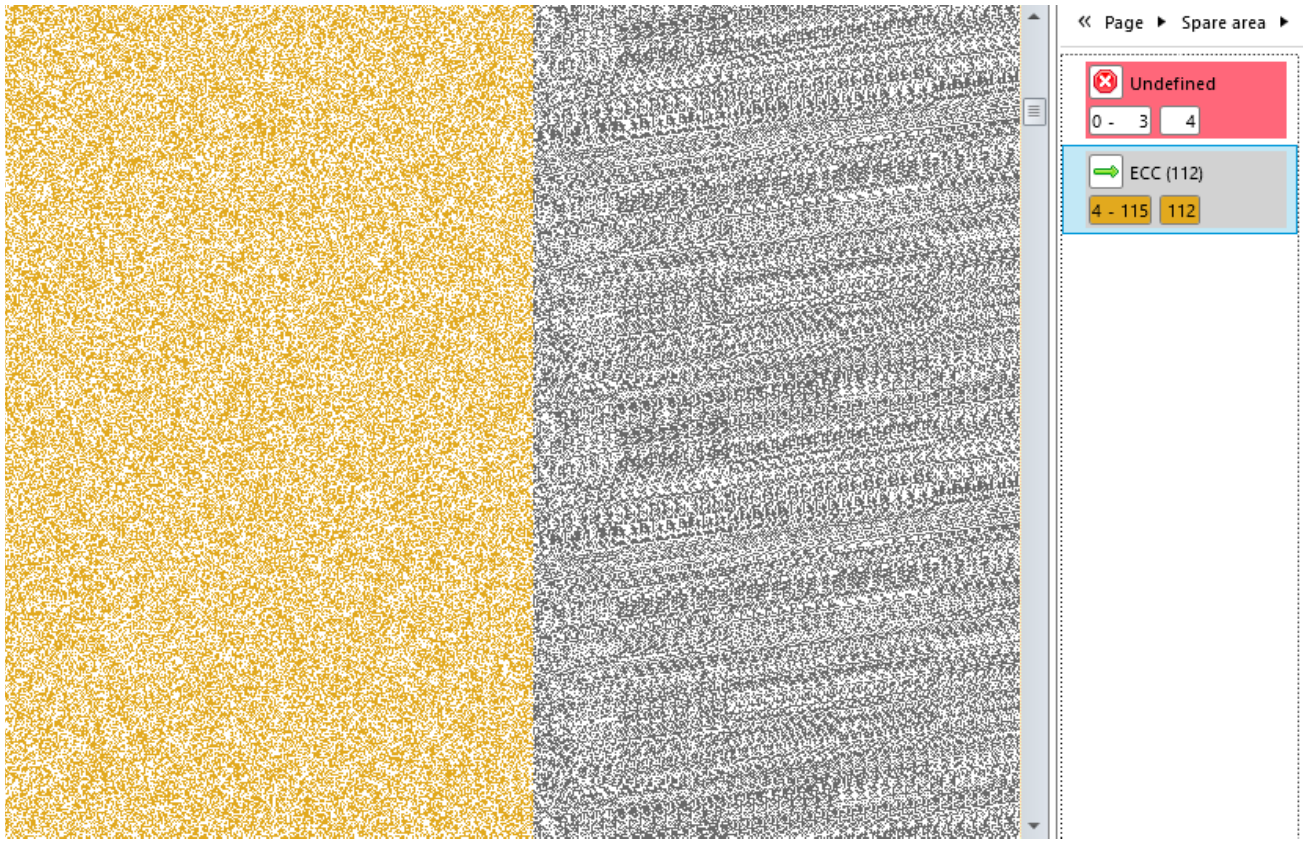
ECC

The ECC area has very high density (entropy). It looks like the noise and has no patterns. Sometimes ECC code has one empty byte (FF) at the end, that looks like the column.





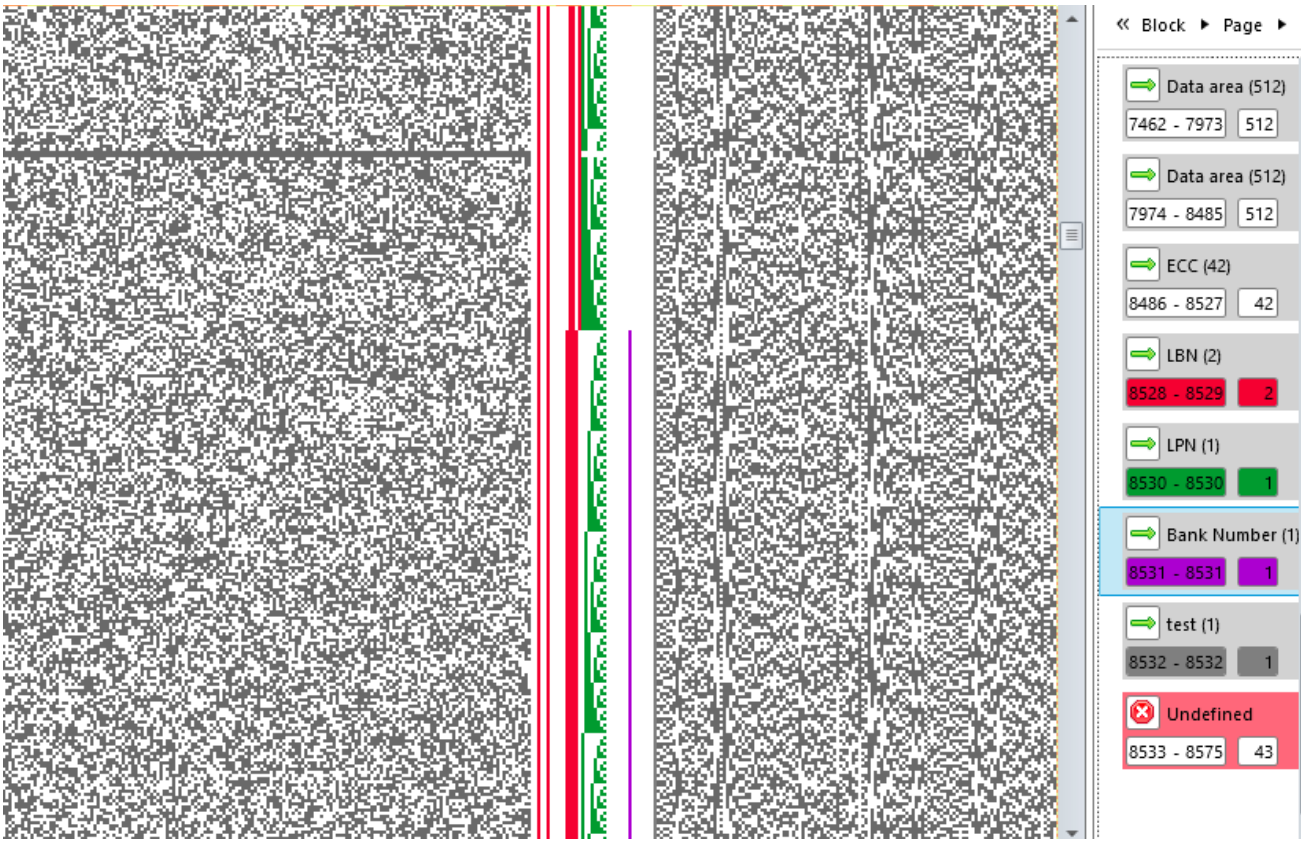




The ECC pattern and it’s offset identification is essential for page structure analysis. The ECC area size determination helps to find ECC decoder manually if automatic analysis doesn’t work. Every Data area (512 or 1024 bytes) has it’s own ECC area.

Bank number

The bank number pattern has 1-byte size and being changed very seldom, because number of banks is limited by 2/4/8/16. It’s represented on the following pictures.



The Bank Number pattern and it’s offset identification is essential for block filtering and sorting in the translation table when building logical image (Markers table element settings). More than 90% of controllers do not use bank number due to sequential bank allocation within physical image.

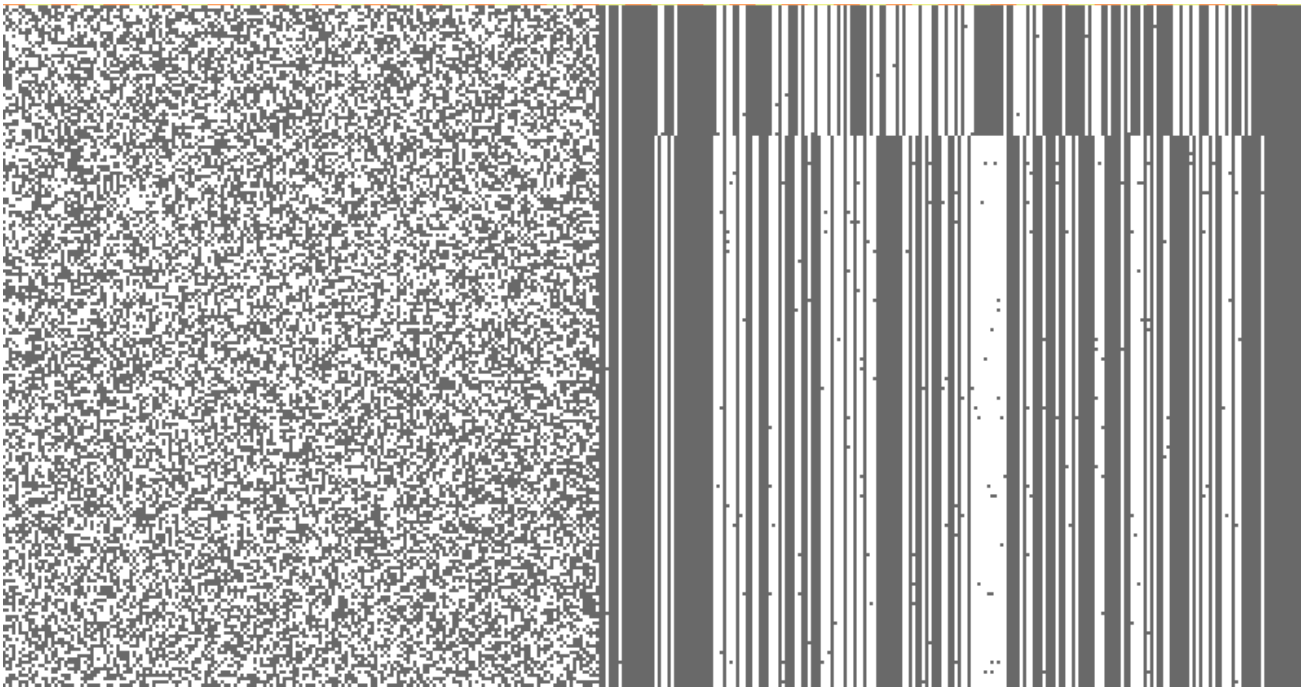
Write counter

The write counter pattern has 1-byte size and being changed from block to block. It may have an interest for digital forensic purposes.

Bit errors

Bit error patterns look like contaminations in the form of “bad pixels”. In the data area of page the bit error estimation is difficult, especially

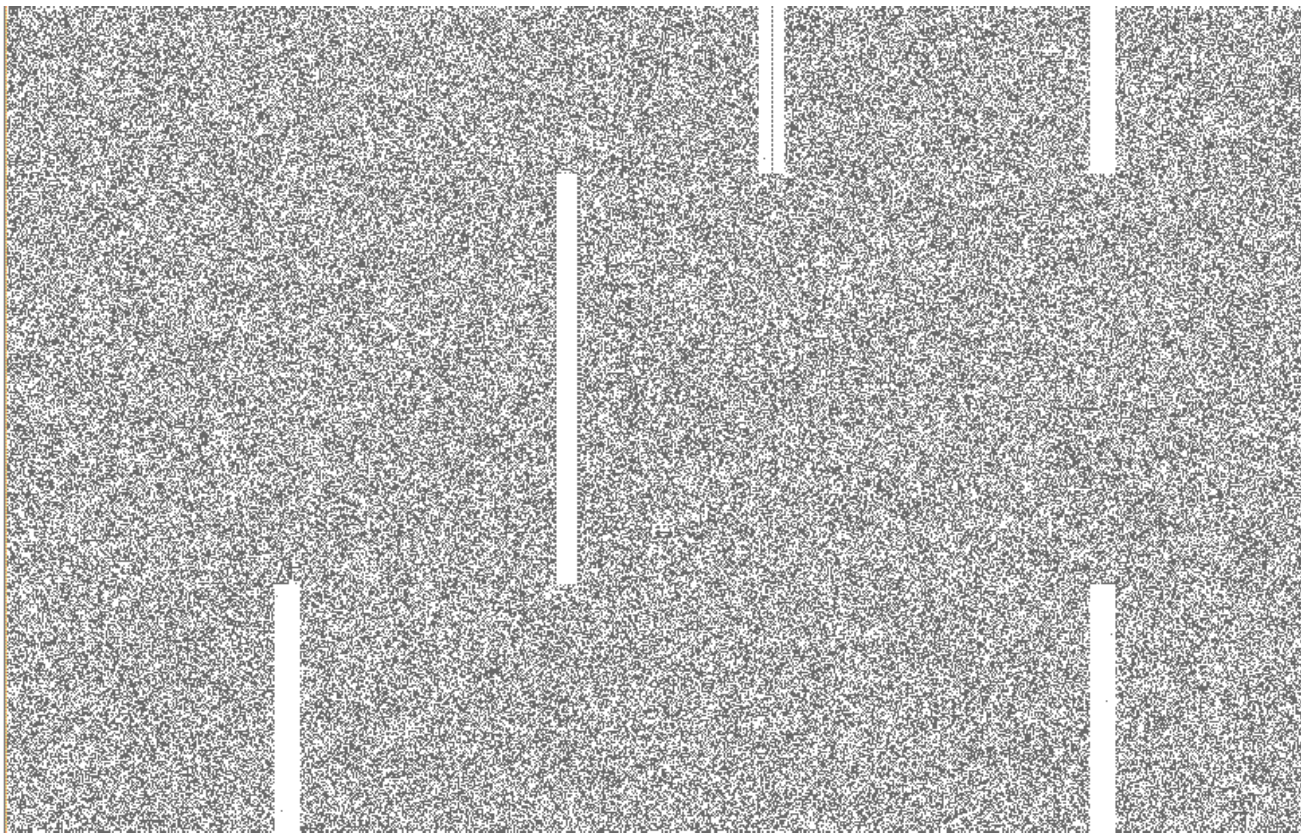
if data scrambled. However, they are clearly visible in the Spare area.



The distribution of bit errors is random. A visual evaluation of number of bit errors helps to determine if power adjustment is needed.

Bad columns

Bad Columns patterns appear as vertical columns with a width of 2 or 1 bytes. They are typically filled FFFF, 0000, or any other value.



The Bad Columns pattern and it’s offsets identification is essentially for factory defects removal from a physical image. The detailed description of the Bad Columns removal procedure can be found in this article.

Challenge 1

1. Examine the structure of Physical image using Bitmap viewer and Structure viewer. Go into page structure and examine page structure. Try to understand patterns and their borders. Analyze the internal spare area structure.
2. Remove all structures except Plane and Page. Determine block size and internal page structure using Bitmap and Structure viewers. Create and assign structures (Block, Page, Data area, Spare area).
3. Analyze spare area structure, find LBN and ECC and assign them inside spare area structure.
4. Check if bit errors exist.

Download the case

Challenge 2

1. Examine the structure of Physical image using Bitmap viewer and Structure viewer. Go into page structure and examine page structure. Try to understand patterns and their borders. Analyze the internal spare area structure.
2. Remove all structures except Plane and Page. Determine block size and internal page structure using Bitmap and Structure viewers. Create and assign structures (Block, Page, Data area, Spare area).
3. Analyze spare area structure, find LBN and ECC and assign them inside spare area structure.
4. Check if bit errors exist.

Download the case